Java 入门- 预科班内容正课会重讲

内参保密不得外传

达内 Java 培优方向

陈子枢 主讲

今日头条极速版

2020

1 第五天：API 工具包：字符串+类型转换+身份证号解析

1.1 常用的 api 包

1.1.1 JDK 包

很多已经过气的技术，被新的框架所替代。

1.2 Object 万物皆对象

1.2.1 默认的 toString 方法

```
        public static void main(String[] args) {
            //类名@地址
            TestArea ta = new TestArea();
            System.out.println(ta);
            System.out.println(ta.toString());
```

```
    }
```

执行结果

打印对象，展示的是对象的内存地址

```
javase.base.TestArea@659e0bfd
javase.base.TestArea@659e0bfd
```

1.2.2 源码

```
public String toString() {
return getClass().getName() + "@" + Integer.toHexString(hashCode());
}
```

1.2.3 覆盖父类的 toString 方法

```
package javase.base;

public class TestArea {
    public static void main(String[] args) {
        String s = new String("chenzishu");
        System.out.println(s.toString());

        //类名@地址
        TestArea ta = new TestArea();
        System.out.println(ta);
        System.out.println(ta.toString());
    }

    @Override  //重写，覆盖父类的 toString 方法
    public String toString() {
        return this.getClass().getSimpleName();
    }
}
```

执行结果:

```
chenzishu
TestArea
TestArea
```

1.2.4 equals

equals 方法是判断两个对象是否相等，因为对象都是引用类型，实际比较的是对象引用所指向的内存地址，==是值比较，大致了解即可

```
package javase.base;

import javase.base.extend.Tiger;

public class TestArea {
    public static void main(String[] args) {
```

```
        //基础类型是按值比较
        int i1 = 10;
        int i2 = 10;
        int i3 = 12;
        System.out.println(i1==i2);
        System.out.println(i1==i3);

        //对象为引用类型，引用类型按内存地址比较
        Tiger t1 = new Tiger();
        Tiger t2 = new Tiger();
        Tiger t3 = t1;

        System.out.println(t1);
        System.out.println(t2);
        System.out.println(t3);

        System.out.println(t1.equals(t2));
        System.out.println(t1 == t2);

        System.out.println(t1.equals(t3));
        System.out.println(t1 == t3);
    }
}
```

执行结果：

```
true
false
javase.base.extend.Tiger@659e0bfd
javase.base.extend.Tiger@2a139a55
javase.base.extend.Tiger@659e0bfd
false
false
true
true
```

评论：【了解】最为基础，应该了解，但实际开发使用并不多。

1.2.5 hashCode

如果两个对象相同， equals 方法一定返回 true，并且这两个对象的 HashCode 一定相同

```
String s = "abc"; //在字符串常量池中分配内存，地址给 s
System.out.println(s.hashCode());

String y = "abc"; //无需分配，字符串共享，地址给 ss
System.out.println(y.hashCode());
```

执行结果：

```
96354
96354
```

1.3 String

1.3.1 转义字符

```
    //在两个字符串中间输入个换行符
    String s1 = "abc\n123";
    System.out.println(s1);

    //在两个字符串中间输入个 TAB 键
    String s2 = "abc\t123";
    System.out.println(s2);
```

1.3.2 concat

拼接字符串

```
    String s1 = "abc";
    String s2 = "def";

    //这里不是数学里的加号，而是字符串的拼接符号
    System.out.println(s1 + s2);

    //concat 对象的拼接方法
    System.out.println("abc".concat("def"));
```

1.3.3 format

以占位符形式展现，代码结构更加清爽

```
    String msg = "";

    Integer i = new Integer(100);
    msg = String.format("%s 的爹是：%s", "整数 Integer",
i.getClass().getSuperclass());
    System.out.println(msg);
```

1.3.4 length

获取字符串的长度

```
    String s = "tony";

    int len = s.length();
    System.out.println(len);
```

评论：【死记】非常常用哦，没什么说的，死记!

1.3.5 trim

截取字符串两边的空格

```
String input = "      不用多久，我就会升职加薪，"
       + "当上总经理，出任  CEO，"
       + "迎娶白富美，走上人生巅峰，想想还有点小激动          ";
System.out.println("未去空格：" + input);
System.out.println("去掉空格：" + input.trim());
```

注意：在“CEO”前面我故意加了一个空格，执行结果可以看到，只把两头的空格去掉，不论多少个，但字里行间的空格是不删除的。

1.3.6 chatAt

获取指定位置的字符

```
String s = "tony";
System.out.println( s.charAt(0) );

String name = "张居正";
System.out.println( name.charAt(0) );
System.out.println( name.charAt(1) );
System.out.println( name.charAt(2) );
```

执行结果：

```
t
张
居
正
```

评论：【了解】用的不多，一般就用于获取首字母。

1.3.7 substring

电话格式符：

```
package javase.base;

public class TestArea {
    public static void main(String[] args) {
        //截取字符串
        String phone = "13572801415";

        String p1 = phone.substring(0,3);
        System.out.println(p1);

        String p2 = phone.substring(3,3+4);
        System.out.println(p2);
```

```
            String p3 = phone.substring(7,7+4);
            System.out.println(p3);

            phone = p1+"-"+p2+"-"+p3;
            System.out.println(phone);    //135-728-01415
        }
    }
```

文件后缀名：

```
    package javase.base;

    public class TestArea {
        public static void main(String[] args) {
            //截取字符串
            String filename = "girl.mp3";
            //得到它的扩展名 mp3
            String extName = filename.substring(5);
            System.out.println(extName);
        }
    }
```

评论：【死记】用的太多了，必须会!

1.3.8 toUpperCase & toLowerCase

```
    String s = "HelloWorld";
    System.out.println("全大写："+ s.toUpperCase() );
    System.out.println("全小写："+ s.toLowerCase() );
    //获取到第一个字母转换成大写字母，在获取后面所有字母，拼串
    System.out.println("首字母小写 ："+ s.toLowerCase().charAt(0) +
s.substring(1, s.length()) );
```

评论：【了解】用的不多，记住即可。

1.3.9 indexOf

查询子串所在位置，返回下标

```
    String mail = "tony@tedu.cn";
    int pos = mail.indexOf("@");
    System.out.println("@所在位置：" + pos);
    System.out.println("邮箱的用户名：" + mail.substring(0,pos));
    System.out.println("邮箱的服务器域名：" + mail.substring(pos+1));
```

执行结果：

```
    @所在位置：4
    邮箱的用户名：tony
    邮箱的服务器域名：tedu.cn
```

评论：【死记】实际开发非常常用，死记。

1.3.10 lastIndexOf

获取字符串最后出现的位置，倒数的位置

```
package javase.util;

public class FileUtils {
    public static void main(String[] args) {
        String fileName = "java 培优+java 高手加薪班宣讲.mp4";
        System.out.println( FileUtils.getExtName(fileName) );
    }

    //获取文件后缀的方法
    public static String getExtName(String fileName) {
        int pos = fileName.lastIndexOf(".");
        String extName = fileName.substring(pos+1);

        return extName;
    }
}
```

注意：上面提前学习了两个陌生的东西，一个工具类方法的创建，一个 static 关键字。先可以放在一边，后面会有章节专门去讲它们的作用，现在先死记。

评论：非常常用，必须熟练！

1.3.11 startsWith & endsWith

根据后缀来判断，前缀雷同

```
String fileName = "HelloWord.class";

if(fileName.endsWith(".java")) {
    System.out.println("这是纯文本文件，称为 java 源代码文件");
}else if(fileName.endsWith(".class")) {
    System.out.println("这是二进制文件，称为 java 编译后的类文件");
}
```

评论：【了解】用的较少，记住即可。

1.3.12 replace & replaceAll

替换 replaceAll 替换所有空格为逗号

```
String str = "java.base.HelloWorld";
System.out.println( str.replace(".", "/"));
System.out.println( str.replaceAll(".", "/"));
System.out.println( str.replaceAll("\\.", "/"));
```

执行结果：

```
java/base/HelloWorld
//////////////////
java/base/HelloWorld
```

为什么这样呢?

注意 replaceAll(String regex) 参数为正则表达式。.在正则表达式中代表一个字符，如何是一般的.的加转义字符\，转义字符在正则中也有特殊含义，所以得写两个。

1.3.13 split

根据分隔符进行分离，存在一个数组中

```
String fullName = "java.base.HelloWorld";
String[] arr = fullName.split("\\.");
for(String s : arr ) {
    System.out.println(s);
}
```

执行结果:

```
java
base
HelloWorld
```

评论：【死记】上面两种实际开发用的最多，一种以逗号作为分隔符，一种以空格作为分隔符。

1.4 Integer

1.4.1 整数转字符串

技巧：java 代码执行一般情况下是从上往下，从左往右。利用""+ 1998，执行时，java 先把 1998 变成字符串类型，然后空串""和 1998 进行字符串连接。下面看到，明显第二行的比第三行代码更加简洁。

```
System.out.println(1998 + 3 + 15);
System.out.println(""+ 1998 + 3 + 15);
System.out.println(String.valueOf(1998) + String.valueOf(3) +
String.valueOf(15));
```

执行结果:

```
2016
1998315
1998315
```

1.4.2 字符串转换为整形

```
Integer i = Integer.parseInt("101");
System.out.println( i );
```

评论：【死记】实际开发这非常常用，死记。

1.5 案例

1.5.1 数字用*替代

```
package javase.string;

import org.junit.Test;

//功能：把数字部分替换*
public class Number2Star {
    @Test  //姓名第二个字替代为*
    public void name() {
        String name = "周星驰";
        String r = name.charAt(0)+"*"+name.substring(2);
        System.out.println(r);
    }

    @Test // 利用字符判断进行替换
    public void charReplace() {
        String s = "tony0123 hellen456 len789";
        String s1 = "";

        for (int i = 0; i < s.length(); i++) {
            char c = s.charAt(i); // 97 a
            // 最小值0 (ASCII 48) , 最大值9 (ASCII 57) , 范围0~9
            //Character c1 = '9';
            //System.out.println((int)c1);

            if (c >= 48 && c <= 57) { // 在0~9，代表是一个数字
                c = '*';
            }
            s1 += c;
        }
        System.out.println(s1);
    }

    @Test // 字符串的替换方法resplace
    public void replace() {
        //替换方式实现把数字变成*
        String s = "tony0123 hellen4056 len7809";
        for(int i=48; i<58; i++) {
            s = s.replace((char)i, '*');
        }
        //r = s.replaceFirst("0", "*");  //regex正则表达式
        System.out.println(s);
    }
```

```
    @Test   //正则表达式实现：\\d 代表 0~9
    public void regex() {
        String s = "tony0123 hellen4056 len7809";
        s = s.replaceAll("\\d", "*");
        System.out.println(s);
    }
}
```

1.5.2 身份证号解析

身份证号里隐含了很多的信息：

1．号码的结构

公民身份号码是特征组合码，由十七位数字本体码和一位校验码组成，共计 18 位。排列顺序从左至右依次为：六位数字地址码，八位数字出生日期码，三位数字顺序码和一位校验码。

2．地址码

表示编码对象常住户口所在县（县级市、旗、区）的行政区划代码，按 GB/T2260 的规定执行。

3．出生日期码

表示编码对象出生的年、月、日，按 GB/T7408 的规定执行，年、月、日代码之间不用分隔符。

4．顺序码

表示在同一地址码所标识的区域范围内，对同年、同月、同日出生的人编定的顺序号，顺序码的奇数分配给男性，偶数分配给女性。

5．校验码

根据前面十七位数字码，按照 ISO 7064:1983.MOD 11-2 校验码计算出来的检验码。

业务复杂吗？业务逻辑复杂吗？复杂就对了，我们不能老玩简单的，这才和实际业务匹配。提升我们的业务处理能力。

```
package javase.base;

import java.util.Arrays;

public class TestArea {
    public static void main(String[] args) {
        // 需求：身份证号码，地区号+生日：年+月+日+性别
        String cardNo = "61011319880315211X";

        String area = ""; // 前 6 位，国家标准（国标 GB）
        // 截串，身份证 18 位，固有长度
        area = cardNo.substring(0, 6);
        System.out.println("区域编码：" + area);
```

```
            String birthday = "";
            birthday = cardNo.substring(6, 14);
            System.out.println("出生年月：" + birthday);

            // 打印中国习惯格式：xxxx 年 xx 月 xx 日
            int year; // 从中间变量 birthday 来截取
            year = Integer.parseInt(birthday.substring(0, 4)); // 字符串转
成整数
            System.out.println(year);

            int month;
            System.out.println(birthday.substring(4, 6));
            month = Integer.parseInt(birthday.substring(4, 6));
            System.out.println(month);

            int day;
            day = Integer.parseInt(birthday.substring(6));
            System.out.println(day);

            // 中国：xxxx 年 x 月 x 日，小技巧：代码执行顺序，从上到下，从左到右
            System.out.println("" + year + month + day);
            System.out.println(year + "年" + month + "月" + day + "日");

            //倒数第二位
            String sex = "";
            sex = cardNo.substring( cardNo.length() -2 , cardNo.length()
-1);
            System.out.println(sex);
        }
    }
```

评论：【死记】用的太多了，必须会!